Findings Reference

Learn what each scan finding means and how to fix it.

Overview

Overview

Overview

Understanding Scan Findings

What Scan Findings Are

Scan findings are the results generated when Steel Security analyzes your WordPress site.

Each finding highlights a potential risk, exposure, or configuration issue that may affect your site's security.

Why Findings Matter

Findings help you understand where your site may be vulnerable.

They are designed to:

- identify potential security risks
- highlight areas of unnecessary exposure
- guide you toward practical improvements

Not all findings indicate an immediate threat, but each represents an opportunity to strengthen your site.

How Findings Are Organized

Steel Security groups findings into logical categories to make them easier to understand and act on.

These categories may include:

- file exposure
- execution risks
- system and information exposure
- endpoint and access risks
- security headers

Grouping findings helps you focus on specific areas of your site.

Understanding Severity

Each finding is assigned a severity level to help prioritize action.

Severity reflects the potential impact and likelihood of exploitation.

Typical severity levels include:

- **High** — significant risk, should be addressed promptly
- **Medium** — moderate risk, should be reviewed and resolved
- **Low** — lower risk, but still worth addressing

Severity helps guide your response, but should not be the only factor in decision-making.

How to Approach Findings

When reviewing findings:

1. start with higher severity items
2. review the context of each finding
3. determine whether the issue applies to your site
4. apply fixes where appropriate
5. test your site after making changes

Avoid applying changes blindly — understanding the impact is important.

Not All Findings Require Action

Some findings may:

- reflect intentional configuration choices
- be acceptable based on your use case
- require a balanced decision between security and functionality

Steel Security provides guidance, but final decisions depend on your environment.

How Findings Connect to Hardening

Each finding typically corresponds to a hardening control.

For example:

- exposed files → file protection controls
- execution risks → execution restrictions
- missing headers → security header controls

This relationship helps you move from detection to resolution.

What to Do Next

After reviewing your findings:

- prioritize based on severity and relevance
- apply appropriate hardening controls
- verify changes after implementation
- continue monitoring over time

Key Principle

Findings are not just warnings.

They are actionable insights that help you improve your site's security posture over time.

Related

- Reviewing Findings
- Hardening Reference
- [Defense in Depth with Steel Security](https://docs.steelsecurity.com/books/hardening-reference/page/defense-in-depth-with-Steel Security)

Overview

How Findings Are Categorized

Why Findings Are Categorized

Steel Security groups findings into categories to make them easier to understand and act on.

Each category represents a different type of security concern.

This structure helps you quickly identify where issues exist within your site.

Main Categories

Findings are organized into the following core categories:

File Exposure

Issues related to files that should not be publicly accessible.

Examples include:

- configuration files
- backup files
- directory listings

Execution Risks

Issues that allow code to run in unintended ways.

Examples include:

- PHP execution in upload directories
- direct access to internal scripts

System & Information Exposure

Issues where your site reveals internal details.

Examples include:

- debug mode enabled
- version information exposed
- system metadata leakage

Endpoint & Access Risks

Issues related to exposed entry points or access controls.

Examples include:

- XML-RPC enabled
- unrestricted access to sensitive endpoints

Security Headers

Issues related to missing or weak browser-level protections.

Examples include:

- missing security headers
- incomplete header configuration

How to Use Categories

Categories help you:

- focus on specific types of risk
- address related issues together
- understand how findings connect to hardening controls

You may choose to resolve findings category by category, or prioritize based on severity.

How Categories Connect to Hardening

Each category aligns with a set of hardening controls.

For example:

- file exposure findings → file protection controls
- execution risks → execution controls
- headers → security header configuration

This alignment makes it easier to move from detection to resolution.

What to Do Next

- review findings within each category
- prioritize based on severity and relevance
- apply corresponding hardening controls

Related

- Understanding Scan Findings
- Severity Levels Explained
- Hardening Reference

Severity Levels Explained

What Severity Means

Severity indicates how important a finding is based on its potential impact and likelihood of exploitation.

It helps you prioritize which issues to address first.

Severity Levels

Steel Security assigns each finding a severity level:

High

High severity findings represent significant security risks.

These issues:

- may expose sensitive data
- may allow unauthorized access or execution
- are more likely to be targeted or exploited

Recommended Action:
Address these findings as soon as possible.

Medium

Medium severity findings represent moderate risks.

These issues:

- may increase your site's attack surface
- may contribute to larger vulnerabilities
- are important but not immediately critical

Recommended Action:
Review and resolve these findings in a timely manner.

Low

Low severity findings represent lower-risk issues.

These issues:

- may involve minor exposure or best-practice gaps
- are less likely to be exploited directly
- still contribute to overall security posture

Recommended Action:
Address these as part of routine maintenance.

How Severity Is Determined

Severity is based on a combination of factors, including:

- potential impact if exploited
- likelihood of exploitation
- level of exposure
- common attack patterns

Severity provides guidance, but does not replace judgment.

Severity Is Not Everything

While severity is important, it should not be the only factor in decision-making.

Consider:

- your specific environment
- whether the behavior is intentional
- how the finding relates to other risks

Some lower severity findings may still be important in your context.

How to Prioritize Findings

A practical approach:

1. address high severity findings first
2. review medium severity findings next
3. resolve low severity findings over time

Also consider grouping related findings for more efficient fixes.

Avoid Overreaction

Not every finding requires immediate action.

Steel Security is designed to surface risks, not create urgency where it is not needed.

Take a measured and informed approach.

Key Principle

Severity helps guide your decisions, but effective security comes from applying multiple protections over time.

Related

- Understanding Scan Findings
- How Findings Are Categorized
- Reviewing Findings

File Exposure Findings

File Exposure Findings

Configuration Files Exposed

What This Means

This finding indicates that one or more configuration files are publicly accessible.

These files may contain sensitive information about your site.

Why It Matters

Configuration files often include critical details such as:

- database credentials
- API keys
- application settings
- internal paths

If exposed, attackers may use this information to:

- gain unauthorized access
- compromise your database
- escalate attacks against your site

How Steel Security Detects This

Steel Security checks for common configuration files that should not be accessible via a browser.

This may include:

- backup copies of configuration files
- misnamed or duplicated config files
- files stored in publicly accessible locations

If a file can be accessed directly, it is flagged.

How to Fix It

To resolve this issue:

- restrict access to configuration files using server rules
- remove unnecessary or duplicate configuration files
- ensure sensitive files are not stored in public directories

You may also use Steel Security hardening controls related to file protection.

What to Expect After Fixing

After applying protections:

- configuration files will no longer be accessible via browser
- requests to these files should return an error (e.g., 403 Forbidden)
- sensitive data will be better protected

How to Verify

To verify the fix:

1. attempt to access the file via its URL
2. confirm that access is denied

Ensure that:

- the file is no longer publicly accessible
- your site continues to function normally

Common Causes

- leftover backup files (e.g., `.bak`, `.old`, `.zip`)
- misconfigured server rules
- manual file uploads to public directories

Best Practices

- never store sensitive files in publicly accessible locations
- remove unused or outdated configuration files
- apply server-level access restrictions
- review file structure regularly

Related

- Protect Configuration Files
- Protect Backup Files
- Limit Exposure of System Info

File Exposure Findings

Backup Files Exposed

What This Means

This finding indicates that backup files are publicly accessible on your site.

These files may contain full or partial copies of your website or database.

Why It Matters

Backup files often include:

- complete site data
- database exports
- configuration information
- user data

If exposed, attackers may:

- download your entire site
- extract sensitive information
- identify vulnerabilities
- gain access to your database structure

How Steel Security Detects This

Steel Security scans for common backup file patterns and locations.

This may include files with extensions such as:

- `.zip`
- `.tar`
- `.gz`
- `.sql`
- `.bak`

If these files are accessible via a browser, they are flagged.

How to Fix It

To resolve this issue:

- remove unnecessary backup files from public directories
- move backups to secure, non-public locations
- restrict access using server-level rules

You may also use Steel Security hardening controls related to file protection.

What to Expect After Fixing

After applying protections:

- backup files will no longer be accessible via browser
- sensitive data will not be exposed publicly
- your site structure will remain secure

How to Verify

To verify the fix:

1. attempt to access the backup file via its URL
2. confirm that access is denied or the file no longer exists

---

Common Causes

- backups stored in the web root or uploads directory
- automated backup plugins saving files in public locations
- leftover files from manual backups

Best Practices

- store backups outside the public web directory
- use secure storage locations (offsite or restricted access)
- regularly clean up old backups
- restrict access to any backup-related directories

Related

- Protect Backup Files
- Protect Configuration Files
- Limit Exposure of System Info

Directory Listing Enabled

What This Means

This finding indicates that directory listing is enabled on your server.

This allows visitors to view the contents of directories when no index file is present.

Why It Matters

When directory listing is enabled, anyone can browse files within a directory.

This may expose:

- sensitive files
- backup files
- configuration files
- internal structure of your site

Attackers can use this information to:

- discover hidden or unlinked files
- identify potential vulnerabilities
- gather intelligence about your environment

How Steel Security Detects This

Steel Security checks whether directories can be accessed and listed publicly.

If a directory returns a file listing instead of an error or redirect, it is flagged.

How to Fix It

To resolve this issue:

- disable directory listing at the server level
- ensure directories do not expose file listings
- apply Steel Security hardening controls related to directory protection

---

What to Expect After Fixing

After disabling directory listing:

- directories will no longer display file contents
- access attempts will return an error or redirect
- internal file structures will remain hidden

How to Verify

To verify the fix:

1. navigate to a directory without an index file
2. confirm that file listings are not displayed

Expected results include:

- a 403 Forbidden response
- a redirect
- or a blank/blocked page

Common Causes

- default server configuration allowing directory listing
- missing index files in directories
- misconfigured hosting environments

Best Practices

- disable directory listing globally

- avoid storing sensitive files in publicly accessible directories
- use proper server configuration for access control
- regularly review directory exposure

Related

- Protect Backup Files
- Protect Configuration Files
- Limit Exposure of System Info

Execution Risks

Execution Risks

Execution Risks

PHP Execution in Uploads

What This Means

This finding indicates that PHP files can be executed within upload or storage directories.

These directories are typically intended for file storage, not code execution.

Why It Matters

Upload directories (such as `/wp-content/uploads/`) are commonly writable by the application.

If PHP execution is allowed in these locations, attackers may:

- upload malicious scripts
- execute code through vulnerable upload points
- gain unauthorized access or control

This is a common method used in real-world WordPress attacks.

How Steel Security Detects This

Steel Security checks whether PHP files placed in upload directories can be executed.

If execution is possible, the directory is flagged as a risk.

How to Fix It

To resolve this issue:

- disable PHP execution in upload and storage directories
- apply server-level rules to prevent script execution
- use Steel Security hardening controls related to execution restrictions

What to Expect After Fixing

After applying protections:

- PHP files in upload directories will not execute
- uploaded scripts will be treated as files, not code
- your site will be significantly more resistant to upload-based attacks

---

How to Verify

To verify the fix:

1. upload or place a test PHP file in an upload directory
2. attempt to access it via browser
3. confirm that execution is blocked

Expected results include:

- file download instead of execution
- or a blocked request (e.g., 403 Forbidden)

Common Causes

- default server configuration allowing execution
- missing directory-level restrictions
- insecure file upload handling

Best Practices

- never allow PHP execution in upload directories
- treat storage locations as non-executable
- regularly review upload behavior and permissions
- combine with other execution and file protection controls

Related

- Prevent Execution in Sensitive Directories
- Block Direct PHP Access
- Restrict Sensitive Endpoints

Execution Risks

Direct PHP Access Allowed

What This Means

This finding indicates that certain PHP files on your site can be accessed directly via a browser.

These files may not be intended to be executed outside of normal WordPress workflows.

Why It Matters

Many PHP files are designed to be included internally, not accessed directly.

If these files are accessible, attackers may:

- execute scripts in unintended ways
- bypass application logic
- probe for vulnerabilities
- trigger unintended behavior

Restricting direct access reduces these risks.

How Steel Security Detects This

Steel Security identifies PHP files that should not be directly accessible and checks whether they can be executed via a browser.

If direct access is possible, the file is flagged.

How to Fix It

To resolve this issue:

- restrict direct access to sensitive PHP files using server rules

- ensure scripts are only executed through WordPress entry points
- apply Steel Security hardening controls related to PHP access restrictions

---

What to Expect After Fixing

After applying protections:

- direct requests to restricted PHP files will be blocked
- unauthorized execution attempts will fail
- normal site functionality should remain unchanged

How to Verify

To verify the fix:

1. attempt to access a flagged PHP file directly via its URL
2. confirm that access is denied (e.g., 403 Forbidden)

Ensure that:

- restricted files are not accessible directly
- legitimate site functionality continues to work

Common Causes

- plugin or theme files exposed without access restrictions
- custom scripts placed in public directories
- missing or incomplete server rules

Best Practices

- avoid exposing internal PHP files directly
- route execution through WordPress where possible
- review custom scripts and endpoints carefully
- combine with other execution controls

Related

- Block Direct PHP Access
- PHP Execution in Uploads
- Restrict Sensitive Endpoints

System & Information Exposure

System & Information Exposure

System & Information Exposure

Debug Mode Enabled

What This Means

This finding indicates that WordPress debug mode is enabled on your site.

Debug mode is intended for development and troubleshooting, not for production use.

Why It Matters

When debug mode is enabled, your site may display:

- error messages
- warnings and notices
- file paths
- system details

This information can be used by attackers to:

- understand your site structure
- identify vulnerabilities
- gain insight into your environment

How Steel Security Detects This

Steel Security checks whether WordPress debug settings are enabled.

This includes:

- debug output being displayed
- debug logging configuration

If debug mode is active in a way that exposes information, it is flagged.

How to Fix It

To resolve this issue:

- disable debug mode in your WordPress configuration
- ensure errors are not displayed publicly
- use logging instead of on-screen output for troubleshooting

You may also use Steel Security hardening controls related to debug settings.

What to Expect After Fixing

After disabling debug mode:

- error messages will no longer be visible to visitors
- sensitive system details will be hidden
- your site will appear more stable and secure

How to Verify

To verify the fix:

1. reload your site and observe for visible errors
2. confirm that warnings and notices are not displayed
3. optionally review logs for captured errors

Common Causes

- debug mode left enabled after development
- troubleshooting changes not reverted
- misconfigured environment settings

Best Practices

- disable debug mode in production environments

- use error logs for troubleshooting instead of display
- separate development and production configurations
- review debug settings regularly

Related

- Disable Debug Mode
- Limit Exposure of System Info
- Hide Version Information

System & Information Exposure

Version Information Exposed

What This Means

This finding indicates that your site is exposing version information for WordPress or related components.

This information may be visible in page source, headers, or other outputs.

Why It Matters

Version information can help attackers identify:

- known vulnerabilities
- outdated software
- specific attack methods

Even if your site is up to date, exposing version details makes it easier to target.

How Steel Security Detects This

Steel Security checks for common locations where version information may be exposed.

This may include:

- meta tags in page output
- response headers
- publicly visible scripts or assets

If version details are detectable, the site is flagged.

How to Fix It

To resolve this issue:

- remove or hide version information from public output
- limit exposure in headers and metadata
- apply Steel Security hardening controls related to version hiding

---

What to Expect After Fixing

After applying protections:

- version information will no longer be easily visible
- your site will be harder to fingerprint
- there should be no impact on functionality

How to Verify

To verify the fix:

1. view the page source of your site
2. inspect response headers
3. check for visible version strings

Confirm that version details are no longer exposed.

Common Causes

- default WordPress behavior exposing version meta tags
- plugins or themes revealing version information
- server headers including version details

Best Practices

- avoid exposing version information publicly
- keep WordPress and plugins updated
- combine with other information exposure controls
- review headers and output regularly

Related

- Hide Version Information
- Limit Exposure of System Info
- Disable Debug Mode

System Information Exposure

What This Means

This finding indicates that your site is exposing system-level information that may reveal details about its configuration or environment.

This information is not always obvious but can be gathered through various outputs and responses.

Why It Matters

System information may include:

- server details
- software versions
- file paths
- configuration hints
- environment data

Attackers can use this information to:

- identify potential vulnerabilities
- tailor attacks to your specific setup
- gain insight into how your site is structured

Reducing this exposure makes your site harder to analyze and target.

How Steel Security Detects This

Steel Security evaluates areas where system information may be exposed.

This may include:

- error messages
- response headers

- page output
- application behavior

If unnecessary or excessive information is visible, it is flagged.

How to Fix It

To resolve this issue:

- suppress detailed error output
- limit server and application information in responses
- reduce visible metadata and system details

You may also use Steel Security hardening controls related to information exposure.

What to Expect After Fixing

After applying protections:

- less system information will be visible publicly
- responses will reveal fewer internal details
- your site will be more resistant to targeted attacks

Functionality should remain unchanged.

How to Verify

To verify the fix:

1. review page output and error behavior
2. inspect response headers
3. check for exposed system details

Confirm that unnecessary information is no longer visible.

Common Causes

- debug or verbose error settings
- default server configurations exposing details
- plugins or themes revealing internal information

Best Practices

- minimize information exposed in production environments
- use logs instead of visible output for debugging
- review headers and responses regularly
- combine with other hardening controls

Related

- Limit Exposure of System Info
- Disable Debug Mode
- Hide Version Information

Endpoint & Access Risks

Endpoint & Access Risks

Endpoint & Access Risks

XML-RPC Enabled

What This Means

This finding indicates that the WordPress XML-RPC interface is enabled and accessible on your site.

XML-RPC allows remote access to WordPress functionality.

Why It Matters

While XML-RPC has legitimate uses, it is commonly targeted by attackers.

It can be used for:

- brute force login attempts
- amplification attacks (e.g., pingback abuse)
- automated access to site functionality

If not required, it increases your site's attack surface.

How Steel Security Detects This

Steel Security checks whether the XML-RPC endpoint (`xmlrpc.php`) is accessible.

If the endpoint responds to requests, it is flagged as enabled.

How to Fix It

To resolve this issue:

- disable XML-RPC if it is not needed
- restrict access to the XML-RPC endpoint
- apply Steel Security hardening controls related to XML-RPC

What to Expect After Fixing

After applying protections:

- XML-RPC access will be blocked or restricted
- automated attacks using XML-RPC will be reduced
- there should be no impact if XML-RPC is not in use

How to Verify

To verify the fix:

1. attempt to access `/xmlrpc.php`
2. confirm that access is blocked or restricted

Common Causes

- default WordPress configuration (XML-RPC enabled by default)
- legacy integrations using XML-RPC
- lack of endpoint restrictions

Best Practices

- disable XML-RPC unless it is required
- restrict access if partial functionality is needed
- monitor for unusual activity targeting XML-RPC
- combine with other endpoint protections

Related

- Restrict XML-RPC
- Restrict Sensitive Endpoints

- Direct PHP Access Allowed

Endpoint & Access Risks

Sensitive Endpoints Accessible

What This Means

This finding indicates that one or more sensitive endpoints on your site are publicly accessible without restriction.

These endpoints may expose functionality that is commonly targeted by automated attacks.

Why It Matters

Sensitive endpoints are entry points into your application.

Examples include:

- `wp-login.php`
- `xmlrpc.php`
- REST API endpoints

If left unrestricted, attackers may:

- perform brute force login attempts
- probe application behavior
- abuse exposed functionality
- automate large-scale attacks

Restricting access reduces unnecessary exposure.

How Steel Security Detects This

Steel Security checks whether commonly targeted endpoints are accessible without restriction.

If these endpoints respond to requests in an unrestricted manner, they are flagged.

How to Fix It

To resolve this issue:

- restrict access to sensitive endpoints
- limit exposure based on usage and requirements
- apply Steel Security hardening controls related to endpoint protection

---

What to Expect After Fixing

After applying protections:

- access to sensitive endpoints will be limited
- automated attack traffic may be reduced
- legitimate functionality should continue to work if properly configured

---

How to Verify

To verify the fix:

1. attempt to access sensitive endpoints directly
2. confirm that access is restricted under expected conditions

Test both:

- authorized access (should work)
- unauthorized access (should be limited or blocked)

Common Causes

- default WordPress exposure of login and API endpoints
- lack of access restrictions
- publicly accessible endpoints without controls

Best Practices

- restrict endpoints that are not required
- limit access where possible
- monitor access patterns
- combine with other access and execution controls

Related

- Restrict Sensitive Endpoints
- XML-RPC Enabled
- Direct PHP Access Allowed

Headers & Browser Security

Headers & Browser Security

Missing Security Headers

What This Means

This finding indicates that one or more recommended security headers are missing from your site's responses.

Security headers help enforce browser-level protections.

Why It Matters

Security headers instruct browsers on how to handle your site's content.

Without them, your site may be more vulnerable to:

- clickjacking attacks
- cross-site scripting (XSS)
- content type confusion
- unintended data exposure

Adding these headers strengthens client-side security.

How Steel Security Detects This

Steel Security checks your site's HTTP responses for the presence of key security headers.

If expected headers are missing, the site is flagged.

Commonly Missing Headers

Examples of important headers include:

- `X-Frame-Options`

- `X-Content-Type-Options`
- `Referrer-Policy`
- Content Security Policy (CSP)

Not all headers are required in every case, but missing core headers increases risk.

How to Fix It

To resolve this issue:

- apply recommended security headers
- use Steel Security hardening controls for header configuration
- implement server-level headers where necessary

What to Expect After Fixing

After applying headers:

- browsers will enforce additional protections
- certain attack vectors will be reduced
- site functionality should remain unchanged in most cases

How to Verify

To verify the fix:

1. open your browser developer tools
2. inspect a page request in the Network tab
3. review response headers

Confirm that expected security headers are present.

Common Causes

- default server configurations without security headers
- missing or incomplete hardening

- lack of browser-level protections

Best Practices

- apply a baseline set of security headers
- review header configuration regularly
- test after applying changes
- expand protections gradually where appropriate

Related

- Security Headers Overview
- X-Frame-Options
- X-Content-Type-Options
- Referrer-Policy
- Content Security Policy (CSP)

Headers & Browser Security

Weak Header Configuration

What This Means

This finding indicates that one or more security headers are present but not configured optimally.

While protections exist, they may be weaker than recommended.

Why It Matters

Security headers are only effective when properly configured.

Weak configurations may:

- provide limited protection
- allow unintended behavior
- fail to fully mitigate common attack vectors

In some cases, a misconfigured header may offer little to no benefit.

How Steel Security Detects This

Steel Security evaluates the values of detected security headers.

If a header is present but does not meet recommended standards, it is flagged.

Examples of Weak Configuration

Examples may include:

- `X-Frame-Options` set too permissively
- missing or overly broad `Referrer-Policy`
- incomplete or overly permissive CSP rules

The exact issue depends on how the header is configured.

How to Fix It

To resolve this issue:

- review the current header configuration
- adjust values to follow recommended best practices
- apply Steel Security hardening controls where available
- refine server-level configuration if needed

What to Expect After Fixing

After improving configuration:

- browser protections will be more effective
- risk from client-side attacks will be reduced
- site functionality should remain stable if properly tested

How to Verify

To verify the fix:

1. inspect response headers using browser developer tools
2. confirm that header values match recommended configurations
3. test site functionality after changes

Common Causes

- default or incomplete header configurations
- overly permissive settings for compatibility
- manual configuration without full validation

Best Practices

- use recommended baseline configurations
- avoid overly permissive settings
- test changes incrementally
- balance security with functionality

Related

- Missing Security Headers
- Security Headers Overview
- X-Frame-Options
- Referrer-Policy
- Content Security Policy (CSP)